

# Примена техника машинског учења у статичкој верификацији софтвера

Лазар Ранковић ????/2016  
Немања Мићовић 1085/2016  
Урош Стегић 447/2016

Математички факултет

12. мај 2017.

# Садржај

Статичка верификација

Машинско учење

Примене МУ у статичкој верификацији

# Мотивација

- Важност процеса верификације софтвера
- Широка примена метода машинског учења

# Методи верификације

- Апстрактна интерпретација
- Проверавање ограничених модела
- Симболичко израчунавање

# Машинско учење

## Увод

- Грађење и употреба алгоритама који генерализују
- Писање програма који се прилагођавају
- Статистичко учење из података

# Машинско учење

## Мотивација

- Алгоритамски нерешиви тешко решиви проблеми
- Задаци које човек лако решава
- Предиктивна анализа

# Машинско учење

## Класификација

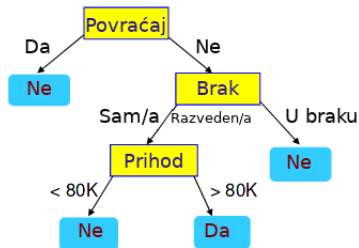
- Препознавање објеката на фотографији
- Разврставање непожељне поште
- Класификација стања програма

# Машинско учење

## Стабло одлучивања

Tid	Povračaj Novca	Bračni status	Prihod	Prevara
1	Da	Sam/a	125K	Ne
2	Ne	U braku	100K	Ne
3	Ne	Sam/a	70K	Ne
4	Da	U braku	120K	Ne
5	Ne	Razveden/a	95K	Da
6	Ne	U braku	60K	Ne
7	Da	Razveden/a	220K	Ne
8	Ne	Sam/a	85K	Da
9	Ne	U braku	75K	Ne
10	Ne	Sam/a	90K	Da

Trening podaci

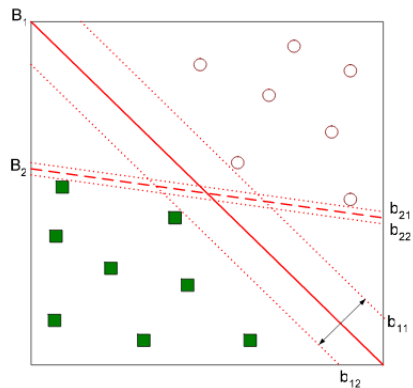


Model: Stablo odlučivanja



# Машинско учење

## Метода потпорних вектора



# Машинско учење

Још мало мотивације

- Магија у НП проблемима
- Машинско учење на белом коњу

## Проналажење интерполанти - потпорни вектори

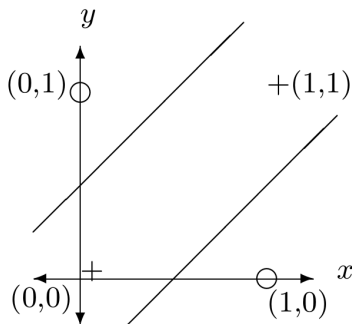
- Делимо скуп стања програма на скупове  $A$  и  $B$
- $A$  и  $B$  се описују формулом логике првог реда
- $A$  садржи вредности  $x$  и  $y$  након линија 1, 2, 3
- $B$  садржи вредности  $x$  и  $y$  након линија 4, 5, 6 и 7

```
funkcija primer()
{
1:   x = y = 0;
2:   while (e)
3:   { x++; y++; }
4:   while (x != 0)
5:   { x--; y--; }
6:   if (y != 0)
7:   greska();
}
```

- Интерполанта је доказ да су скупови  $A$  и  $B$  дисјунктни
- Доказивач теорема рачуна вредности за променљиве  $x$  и  $y$
- Добија се скуп инстанци над којима се може тренирати модел машинског учења

## Проналажење интерполанти - потпорни вектори

- Модел добијен применом метода потпорних вектора
- Приказане праве одговарају једначинама:  
 $p1 : 2y = 2x + 1$   
 $p2 : 2y = 2x - 1$
- Можемо извести интерполанту  
 $2y \leq 2x + 1 \wedge 2y \geq 2x - 1$
- Инваријанта  $x = y$  се може добити транслирањем правих што ближе позитивним инстанцама



# Грађење класификатора нетачне инваријанте

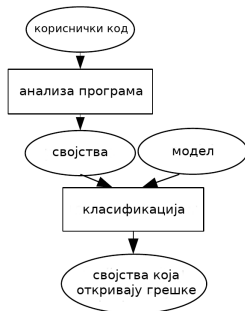
- Рангирају се својста програма по вероватноћи за стварање грешке
- Циљ је програмеру понудити листу својстава које потенцијално треба проверити

```
// Vraca sortiranu kopiju argumenta
double[] bubble_sort(double[] in) {
    double[] out = kopiraj_niz(in);
    for (int x = out.duzina - 1; x >= 1; x--)
        // donja granica treba da bude 0, ne 1
        for (int y = 1; y < x; y++)
            if (out[y] > out[y+1])
                razmeni(out[y], out[y+1])
    return out;
}
```

Својства	Открива грешку?
$out[1] \leq in[1]$	Да
$\forall i: in[i] \leq 100$	Не
$in[0] = out[0]$	Да
$size(out) = size(in)$	Не
$in \subseteq out$	Не
$out \subseteq in$	Не
$in \neq null$	Не
$out \neq null$	Не

## Грађење класификатора нетачне инваријанте

- Користи се DAIKON динамички детектор инваријанти који може да проверава:
  - уређење ( $x \leq y$ )
  - опсег ( $a \leq x \leq b$ )
  - линеарне везе ( $z = ax + by + c$ )
  - и друге
- Добијена својства се кодирају у векторе и примењује се метода машинског учења



Својства	Једначина				Тип променљиве			# v	Резултат
	$\leq$	$=$	$\neq$	$\subseteq$	int	double	array		
$\text{out}[1] \leq \text{in}[1]$	1	0	0	0	0	1	0	2	19
$\forall i: \text{in}[i] \leq 100$	1	0	0	0	0	1	0	1	16
$\text{in}[0] = \text{out}[0]$	0	1	0	0	0	1	0	2	15
$\text{size}(\text{out}) = \text{size}(\text{in})$	0	1	0	0	1	0	0	2	13
$\text{in} \subseteq \text{out}$	0	0	0	1	0	0	1	2	12
$\text{out} \subseteq \text{in}$	0	0	0	1	0	0	1	2	12
$\text{in} \neq \text{null}$	0	0	1	0	0	0	1	1	10
$\text{out} \neq \text{null}$	0	0	1	0	0	0	1	1	10
Тежине модела	7	3	2	1	4	6	5	3	

# Закључак

- Машинско учење итекако може имати примену у верификацији софтвера
- Добијени резултати су упоредиви а негде и бољи од традиционалних метода

Хвала на пажњи